Technical Note	1969-34
Automatic Microwave Circuit Analysis with GCP-MOD2	W. J. Getsinger T. E. Maggiacomo 10 July 1969

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Lexington, Massachusetts

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LINCOLN LABORATORY

AUTOMATIC MICROWAVE CIRCUIT ANALYSIS
WITH GCP-MOD2

W. J. GETSINGER
T. E. MAGGIACOMO

Group 46

TECHNICAL NOTE 1969-34

10 JULY 1969

LEXINGTON   MASSACHUSETTS

ABSTRACT

GCP-MOD 2 is an extremely rapid means for performing frequency analyses of equivalent circuits of cascaded two-ports. An operator describes his circuit to GCP-MOD 2, using an abbreviated plain English vocabulary, and specifies measurements to be made at an appropriate terminal plane for frequencies of interest. GCP communicates laconically with the operator as it accepts the circuit description and any other instructions. It then tabulates the results of the measurements, and awaits further instructions, such as circuit modifications or different measurements.

As a typical example, a microwave filter circuit, made up of lengths of transmission line and five resonators, of both lumped and distributed elements, can be described to GCP and evaluated at a dozen frequencies for such measurements as insertion loss, transmission phase-shift, and input impedance, all in about 10 minutes.

CONTENTS

AUTOMATIC MICROWAVE CIRCUIT ANALYSIS WITH GCP-MOD 2

I. DESCRIPTION OF GCP-MOD 2

A. Introduction

GCP-MOD 2 performs frequency analysis on equivalent circuits of cascaded linear two-ports. It does this by generating the transfer matrices of two-ports described to it, and multiplying them to arrive at an overall transfer matrix for the cascade. The elements of the final matrix are manipulated and operated on to determine "measured" quantities at each frequency.

To work with GCP, an engineer describes his circuit using abbreviated plain-English words given in the SECTION MENU. He directs that various measurements be made on the circuit, using words from the MEASUREMENT MENU. He may change the circuit, tune elements in the circuit, set up additional circuits, call for measurements, and terminate operation by using appropriate words from the list of COMMANDS.

The philosophy on which the design of GCP is based is that the typical user is a research-oriented engineer, rather than a full-time microwave-circuit designer, and that he may use GCP rather infrequently, but intensely. It is assumed that the user does not want to sacrifice much time in learning or relearning to use GCP. On the other hand, he does not want to sacrifice speed or flexibility of operation for a large amount of instructional information from GCP itself.

The compromise reached is a vocabulary of words of up to four letters, which are either truncated English-language words or other mnemonic abbreviations. The four-letter maximum was chosen on the basis of giving the most information for the least redundancy, without using excessive typewriter printing time.

Another consideration is the relative control of the action; is GCP controlling the course of action, or is the user in command? This is another compromise situation, but the COMMAND LIST gives the user options to redirect the course of analysis while GCP controls the details of the specific action chosen by the user.

B. Describing a Circuit

1. Numbering Branches and Sections

The circuit to be described must be capable of representation as a cascade of two-ports. Allowable two-ports, termed "sections" are given in the SECTION MENU.

In setting up a cascade of sections, it is necessary to observe the following system of numbering, in order that each section used have the proper location with respect to others.

1

Each cascade setup is termed a "branch," and is identified by a two-digit number: 01, 02, ..., 09, 10. A maximum of 10 separate branches is permissible at present.

Each section setup is identified by the number of the branch of which it is a part, and another two-digit number giving its position in the cascade. For example, 01, 05 refers to the fifth section in the first cascade. A maximum of 50 sections is permissible in each branch.
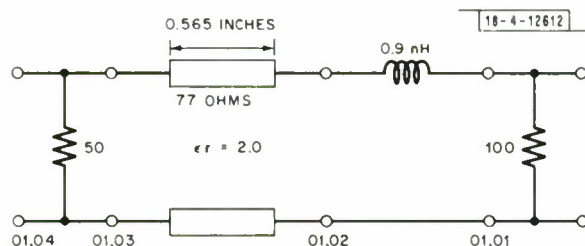

Fig. 1. Example of section numbering.

In consideration of the convention that a generator is on the left, and a load is on the right, the numbering proceeds from right to left, and includes the load, if any. Figure 1 illustrates the numbering.

The terminals on the left of each section bear the same designating number as the section. This convention is used in specifying the port at which a measurement is to be made.

GCP indicates its readiness to operate by printing 01, 01. The user then types the name of the first section of the first branch, preceding the name by SHUN or SERI if the section is made of a two-terminal element connected in shunt or series to form a two-port. The names and descriptions of sections are found in the SECTION MENU.

The example below shows how branch 01 of the preceding illustration is described to GCP:

GCP	USER
EXECUTION BEGINS	
01, 01	shun, resi
RESI (OHMS)	100
01, 02	seri, indu
INDU (NH)	.9
01, 03	teml
ZO, (LENG (IN), DIEL)	77.565, 2.00
01, 04	shun, resi
RESI (OHMS)	50
01, 05	

No voltage or current sources are used with GCP-MOD 2, because it is limited to the analysis of linear two-ports. Driving point and transfer ratios are adequate to characterize the performance of such circuits. For this reason, it is equally valid to number sections within a branch from left to right or from right to left as shown in the illustration. However, care must be taken in evaluating measurements and in using nonreciprocal sections, since right to left numbering has been assumed in developing GCP.

2. Use of Operator Sections

At the end of the SECTION MENU are the OPERATOR SECTIONS. These are special operations occupying one section of a branch. First given are SHUN JUNC and SERI JUNC which connect a side branch (with its own branch number) to the main branch, in a shunt or series connection, as desired. It is to be noted that it is the side branch end having the

2

higher section number that is connected at the main branch. The lower section number end (p, 00) must always remain unconnected. Thus, side branches may not connect to other side branches, nor return to the main branch. That is, they may not form closed loops in the circuit topology. However, a side branch may have one or more subsidiary branches, but these branches-off-branches may not branch further. It is important to note that any set of measurements may be made only within a single branch. Thus, branches may not be connected end-for-end, nor may transfer measurements be made between a generator in one branch and a load in another.

One main branch may have many side branches. If identical side branches are needed, a single side branch may be made up and specified at each junction.

A special pair of operator sections allows a side branch to rejoin the main branch, so that the two run parallel. This is illustrated in OPERATOR SECTIONS. A parallel branch is initiated by the command PARB p, where p is the two-digit branch number of the side branch. Farther along the main branch, the side branch is returned by using a section called COMB, for combine. Observe that the lower-numbered section ends of the two branches involved are connected together in parallel at PARB, and the higher-numbered ends are connected in parallel at COMB. Thus, the branches not only run parallel, they are also in a parallel connection at each end.

A series connection is possible, however, at either or both ends. Section II-G-2 describes the method for making series connections of parallel branches.

Another operator section is IMAG, for image, which allows the user to select any cascade of sections in a branch, and duplicate it in reverse farther along the branch. This is a valuable capability for multi-element filters, which are often symmetrical about the center of the structure. The duplicated part of the cascade occupies only one section, even though the original occupied a number of sections.

C. Physical Units

As set up at present, GCP-MOD 2 uses the following set of units because they are convenient at microwave frequencies:

Length	inches
Frequency	GHz (10^9 Hz)
Impedance	ohms
Admittance	mhos
Inductance	nH (10^{-9} H)
Capacitance	pF (10^{-12} F)

D. Making Measurements

When the operator wishes to have measurements made on the circuit he has described to GCP, he types MEAS. GCP responds by printing BR, SE, TYPE, thus asking the operator for branch and section numbers of the terminal plane at which the measurement is to be initiated, and the type of measurement, given by the name in the MEASUREMENT MENU. The operator

may specify up to three types of measurements to be made, provided all are initiated at the same terminal plane.

There are three basic types of measurements. First, there are driving point measurements of impedance or admittance, which can be made at any terminal plane looking in either direction. Next, there are driving point measurements which must be made adjacent to a purely resistive termination in order to provide a reference impedance level. This includes such measurements as VSWR and reflection loss. Finally, there are transfer measurements, such as transducer loss, or transmission phase shift, which require reference impedance levels at both ends of the overall cascade. Thus, transfer measurements are made only on circuits having resistance terminations at both ends, at terminal planes adjacent to the terminations. It is necessary to specify only the terminal plane assumed to be the generator port for that measurement. GCP assumes the other termination is the load for that measurement.

To return to the circuit example, assume that the operator wishes to measure input admittance, VSWR, and transducer loss for branch 01 of the illustration, assuming the generator port to be 01, 03. The GCP user interaction is as follows:

GCP	USER
	meas
BR, SE, TYPE	01, 03, ylod, vswr, trlo
FREQ, (MIN, MAX, INCR)	4, 5, 0.2

FREQ (GHZ)	ADMITTANCE		VSWR	TRAND LOSS (DB)
4.0000	0.0180	0.0025	1.1861	0.0316
4.2000	0.0186	0.0017	1.1196	0.0139
4.4000	0.0190	0.0006	1.0611	0.0038
4.6000	0.0192	− 0.0005	1.0501	0.0026
4.8000	0.0191	− 0.0018	1.1064	0.0111
5.0000	0.0187	− 0.0030	1.1808	0.0300

NEXT

Measurement of individual branches may be made at any time, but when a circuit has side branches (using JUNC) or has parallel branches (using PARB and COMB), measurements for the entire circuit may be made only on the main branch. When GCP operates on the main branch, it "sees" the side or parallel branches, but does not "see" the main branch when operating on a side or parallel branch.

MEASUREMENT DEFINITIONS are given in Sec. II-C.

E.   Use of Commands

1.   Normal Commands

The action of GCP-MOD 2 can be directed by the operator using the command words given in the COMMAND LIST.

ELMO and SEMO allow modification of an element-value and a section-type, respectively.

BRAN n, where n is the two-digit cascade number, allows the operator to work with any particular branch called. More than one branch might be needed for any of the following reasons:

The operator may wish to make a side calculation, or to try out a subcircuit before incorporating it in the main branch, or the main branch may have junctions connecting to other branches.

The command MEAS directs GCP to accept instructions taken from the MEASUREMENT MENU described previously.

The command TUNE is used when the operator wishes to observe the changes in some measured quantity as the values of some circuit parameter are changed. For example, the transducer loss of a filter may be observed at a critical frequency as the length of a coupled stub is varied over some range of values. In TUNE, only one element may be varied, and up to three types of measurements may be made at only one frequency. The command OFFL is used to print the measured data on an offline printer. The data are written in a file called GCP DATA. After terminating GCP with a QUIT command, the CMS command offline print gcp data should be issued.

OFFL stays in effect until the command ONLI is issued. ONLI will resume the typing of measured data on the console.

The command QUIT is used to terminate operation of GCP. The information or data of the circuits worked with are not erased by this command. Thus, the data of the previous operation are available when GCP-MOD 2 is turned on again. To reinstate this data, it is necessary to use the command RESE, for reset, when GCP prints 01, 01. If a new section is entered when 01, 01 is printed, the file of previous data is erased. RESE is also used when GCP is shut down inadvertently during an operating session. In this case, it should be noted that RESE returns GCP to the next to last branch previously used, rather than to the last section. This is done because shutdown may have occurred after the last section had been entered, but before it had been properly filed. Thus, after RESE, it is necessary to re-enter the last section of the branch involved.

When GCP has completed a given task, it prints NEXT, to indicate it is ready to accept new instructions.

2. Diagnostic Commands

There are two basic diagnostic commands by means of which the user may inspect the data on which GCP is operating. The first of these is LIST, which causes GCP to print out the section types and element values for all branches and sections, or only specified ones. The numbers printed out under LIST can be identified by referring to Sec. II-D under "LIST". The other diagnostic is MATR, which causes GCP to print out the values of the transfer matrix elements (at the last frequency of measurement) of all or some of the sections used in that session.

F. Options

Different areas of application may require different SECTION MENUS and MEASUREMENT MENUS. The generation of special menus is left to the user.

Anticipated additions to GCP will provide graphical output, and an optimization capability.

G. Operating Techniques

1. Techniques

A typical design task involves evaluation of the performance of a proposed circuit, comparison of the performance with design goals, and modification of the circuit in hopes of improving

its performance. The cycle of evaluation, comparison, modification, and re-evaluation is repeated until a realistic circuit having satisfactory performance has been achieved, or found to be unobtainable.

Thus, in the interest of efficient and effective use of GCP, the following technique is offered (introduce no more uncertainty in the stated problem than is absolutely necessary!):

If the circuit can be broken down into subcircuits which can be optimized separately, then these subcircuits should be worked on separately.

If possible, start with a circuit of known response that can be converted into the actual circuit, then measure, modify and optimize one element at a time. For example, consider a filter, bound by physical constraints, to be made in part of lumped elements, and in part of distributed elements. Start with an entirely lumped element prototype having the performance required, since lumped element filters have been completely determined by analytical means, and their properties published widely. Then, investigate the change in performance when only one lumped element section is replaced by the required distributed section. Optimize performance for this single change before going on to the next.

With some problems, it is possible to start with a simple but meaningful approximation to the actual circuit, using EMPT for sections which will later be required to incorporate parasitics. Get the simple approximation to work, and then add the parasitics one at a time, optimizing at each step.

In the interest of simplicity and speed, the Operator Command IMAG is included to duplicate a cascade of sections for a symmetrical network. Observe that the mirror image of many sections can be duplicated using only one section.

Notice, also, that if a number of identical branches is needed in a circuit, it is only necessary to define that branch once, and then specify it by the one branch number at each junction (JUNC).

2. Series Connection for a Parallel Branch

The pair of commands PARB and COMB permits the parallel connection of a branch running parallel to the main branch. The connection at either or both ends can be made a series connection by the following artifice. Each of the three proper circuit sections, which would normally connect to PARB or COMB for a parallel connection, must instead be required to be in cascade with a section-type INTE. This makes a series junction because INTE interchanges voltage and current. PARB or COMB, which in effect add currents to make a parallel connection, will then be adding what are called currents, but which are actually voltages, thus forming a series connection.

3. Transmission Line Loss

GCP does not have a section-type for transmission lines with loss, but loss in transmission lines or waveguide can be handled accurately by the following method.

If we take the transfer matrix for lossy transmission line,

$$\begin{bmatrix} \cosh\gamma\ell & ZO\ \sinh\gamma\ell \\ YO\ \sinh\gamma\ell & \cosh\gamma\ell \end{bmatrix}$$

where $\gamma = \alpha + j\beta$, the complex propagation factor, and ℓ is the length of line, and either pre-multiply or post-multiply by the inverse matrix of a lossless transmission line,

$$\begin{bmatrix} \cos\beta\ell & -jZO\,\sin\beta\ell \\ -jYO\,\sin\beta\ell & \cos\beta\ell \end{bmatrix}$$

which means that we are removing a lossless line from either end, we are left with

$$\begin{bmatrix} \cosh\alpha\ell & ZO\,\sinh\alpha\ell \\ YO\,\sinh\alpha\ell & \cosh\alpha\ell \end{bmatrix} .$$

This is the transfer matrix for a zero-length attenuator of characteristic impedance ZO, and loss of $L = 8.686\,\alpha\ell$ decibels. For a typical transmission line we can compute or measure both ZO and $\alpha\ell$, or L(dB). Thus a lossy line can be treated as a lossless line with a matched attenuator at one end or the other, as convenient.

4. Saving a GCP Session for Future Use

If GCP is inoperative for a period of time, overnight for example, the previous data are stored, and can he recalled by typing RESE in response to GCP typing 01, 01. If, however, it is desired to store data from a particular session without disabling GCP for other use, then it is necessary to ALTER the file GCP STORAGE to another file name and filetype, or have it printed out on cards or tape. The original data may be reinstated with GCP by subsequently renaming the second file back to GCP STORAGE, and calling RESE.
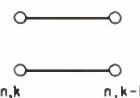
5. User-Generated Sections

Special dummy sections (DUMA, DUMB, DUMC, DUMD) have heen incorporated in the SECTION MENU to allow for user-generated special sections. These are difficult to implement without a thorough knowledge of the internal details of GCP.
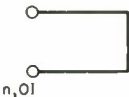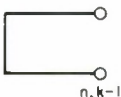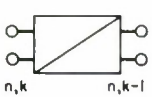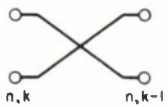
II. MENUS
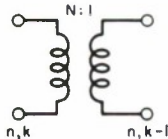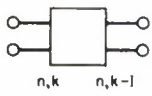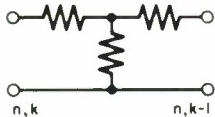
    A. Section MENU

        1. Circuit Sections (n is the two-digit branch number, and k is the two-digit number of the section to be introduced)

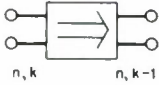ELEMENTARY SECTIONS

GCP NAME AND SYMBOL	GCP WILL REQUEST	TYPE OF SECTION	TRANSFER MATRIX
EMPT	−	Straight through connection occupying one section	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

n,k   n,k−I

GCP NAME AND SYMBOL	GCP WILL REQUEST	TYPE OF SECTION	TRANSFER MATRIX
SHOR	–	Short circuits load end of cascade	
SHOR	–	Short circuits generator end of cascade	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
INTE	–	Interchanges voltage and current	
CROS	–	Inverts polarity of terminals	$\begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$
XFMR	TURNS RATIO	Ideal transformer	$\begin{bmatrix} N & 0 \\ 0 & 1/N \end{bmatrix}$
INVE	ZO(OHMS)	Ideal imittance inverter with 90° phase shift	$\begin{bmatrix} 0 & jZO \\ jYO & 0 \end{bmatrix}$
ATTN	ZO, LOSS(DB)	Ideal attenuator	$\begin{bmatrix} \frac{1}{2}\left(L+\frac{1}{L}\right) & ZO\left(L-\frac{1}{L}\right) \\ YO\left(L-\frac{1}{L}\right) & \frac{1}{2}\left(L+\frac{1}{L}\right) \end{bmatrix}$

$L = 10^{\text{LOSS(DB)}/20}$

The symbols carry labels: SHOR (n,0); SHOR (n,k-1); INTE (n,k) (n,k-1); CROS (n,k) (n,k-1); XFMR N:1 (n,k) (n,k-1); INVE (n,k) (n,k-1); ATTN (n,k) (n,k-1).

8

GCP NAME AND SYMBOL	GCP WILL REQUEST	TYPE OF SECTION	TRANSFER MATRIX
ISOL n,k n,k-1	ZO, LOSS, REVL(DB)	Ideal isolator	$$\begin{bmatrix} \frac{1}{2}(LF + \frac{1}{LR}) & \frac{ZO}{2}(LF - \frac{1}{LR}) \\ \frac{YO}{2}(LF - \frac{1}{LR}) & \frac{1}{2}(LF + \frac{1}{LR}) \end{bmatrix}$$

$$LF = 10^{LOSS(DB)/20} \quad LR = 10^{REVL(DB)/20}$$

SECTIONS MADE OF TWO-TERMINAL CIRCUITS

SHUN NAME where NAME stands for GCP name of two-terminal circuit connected in shunt to formation n, k	Info needed for circuit named	Admittance in SHUNT	$$\begin{bmatrix} 1 & 0 \\ Y & 1 \end{bmatrix}$$
SERI NAME where NAME stands for GCP name of two-terminal circuit connected in series to form section n, k	Info needed for circuit named	Impedance in SERIES	$$\begin{bmatrix} 1 & Z \\ 0 & 1 \end{bmatrix}$$
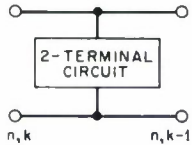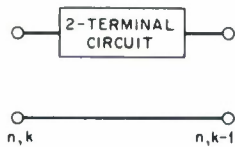
LUMP CIRCUIT ELEMENT SECTIONS

(SHUN or SERI must precede NAME of two-terminal circuits)

GCP NAME AND SYMBOL	GCP WILL REQUEST	TYPE OF SECTION	IMMITTANCE
RESI	RES (OHMS)	Resistance	$Z = R$

GCP NAME AND SYMBOL	GCP WILL REQUEST	TYPE OF SECTION	IMMITTANCE
INDU	IND (NH)	Inductance	$Z = j\omega L$
CAPA	CAP (PF)	Capacitance	$Y = j\omega C$
SRLC	RES (OHMS), IND (NH), CAP (PF)	Series resonant	$Z = R + j(\omega L - \dfrac{1}{\omega C})$
SRES	FO(GHZ), X, 1/Q	Series resonant	$Z = \dfrac{X}{Q} + jX(\dfrac{F}{FO} - \dfrac{FO}{F})$

where

FO = Resonant freq.

$x = 2\pi FOL$, slope parameter

$Q = 2\pi FOL/R$

| PRLC | RES (OHMS), IND (NH), CAP (PF) | Parallel resonant | $Y = \dfrac{1}{R} + j(\omega C - \dfrac{1}{\omega L})$ |
| PRES | FO(GHZ), BP, 1/Q | Parallel resonant | $Y = \dfrac{BP}{Q} + jBP(\dfrac{F}{FO} - \dfrac{FO}{F})$ |

where

FO = Resonant freq.

BP = $2\pi FOC$, slope parameter

$Q = 2\pi FOCR$

TRANSMISSION LINE TWO-PORT SECTIONS
(SHUN or SERI must precede NAME of two-terminal circuits)

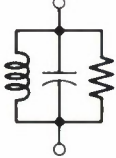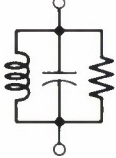| GCP NAME AND SYMBOL	GCP WILL REQUEST	TYPE OF SECTION	IMMITTANCE
TEMS	ZO, LENG (IN.), DIEL	Short-circuited TEM transmission line	$Z = ZO \tan \dfrac{2\pi F(LENG)(DIEL)^{1/2}}{c}$ DIEL = Relative dielectric constant
TEML	ZO, LENG (IN.), DIEL	Open circuited TEM transmission line	$Y = jYO \tan \dfrac{2\pi F(LENG)(DIEL)^{1/2}}{c}$
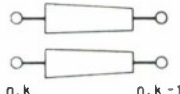
GCP NAME AND SYMBOL	GCP WILL REQUEST	TYPE OF SECTION	TRANSFER MATRIX
TEML n,k n,k-1	ZO, LENG (IN.), DIEL	TEM transmission line (lossless)	$\begin{bmatrix} \cos\Theta & jZO\sin\Theta \\ jYo\sin\Theta & \cos\Theta \end{bmatrix}$ where $\Theta = \dfrac{2\pi F(LENG)(DIEL)^{1/2}}{c}$ and $c = 11.8$ inches/nsec
RADL n,k n,k-1	DLOD, DGEN, B, DIEL DLOD = DIA, Nearest Load, DGEN = DIA, Nearest GEN, B = Spacing of Plates		See Appendix.
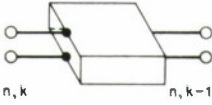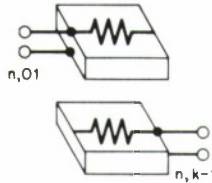
11

WAVEGUIDE ELEMENT SECTIONS[*]

GCP NAME AND SYMBOL	GCP WILL REQUEST	TYPE OF SECTION	TRANSFER MATRIX
TEWG n,k n,k-1	WIDE, NARO, LENG, DIEL	TE-mode rectangular waveguide	$\begin{bmatrix} \cos\Theta & jZO\sin\Theta \\ jYO\sin\Theta & \cos\Theta \end{bmatrix}$ $\Theta = \dfrac{2\pi(F^2 - FC^2)^{1/2}(LENG)(DIEL)^{1/2}}{c}$ $FC = \dfrac{c}{2(WIDE)(DIEL)^{1/2}}$ $ZO = \dfrac{240\pi(NARO)F}{(WIDE)(DIEL)^{1/2}(F^2 - FC^2)^{1/2}}$
WLOD n,01 n,k-1	WIDE, NARO, DIEL	Waveguide termination	$\begin{bmatrix} 1 & 0 \\ YO & 1 \end{bmatrix}$ $YO = 1/ZO$ of TEWG.
SHUN SERI } TEWS	WIDE, NARO, LENG, DIEL	Short-circuited TE10 waveguide	$Z = jZO\tan\Theta$

[*] The definition of characteristic impedance in column 4 is correct for coupling to a probe[†] or packaged diode[‡] mounted across the narrow dimension.   It is also suitable for cascaded waveguides having different dielectrics or different narrow dimensions (except for fringing capacitance).[§]

For cascaded waveguides of differing wide dimensions, both an impedance transformation and fringing susceptance must be accounted for.[¶]

[†] S. A. Schelkunoff, Electromagnetic Waves (D. Van Nostrand, Inc., Princeton, New Jersey, 1943), Sec. 12.4, pp. 494-496.

[‡] W. J. Getsinger, "The Packaged and Mounted Diode as a Microwave Circuit," IEEE Trans. on Microwave Theory and Techniques, MTT-14, 58-69 (February 1966).

[§] N. Marcuvitz, Waveguide Handbook, Rad. Lab. Series (McGraw-Hill, New York, 1951), Vol. 10, Sec. 5.26, pp. 307-310.

[¶] Ibid., Sec. 5.24, pp. 296-302.

GCP NAME AND SYMBOL	GCP WILL REQUEST	TYPE OF SECTION	TRANSFER MATRIX
WISO n,k n,k-1	WIDE, NARO, DIEL, LOSS, REVL(DB)	Waveguide isolator	$$\begin{bmatrix} \frac{1}{2}\left(LF + \frac{1}{LR}\right) & \frac{ZO}{2}\left(LF - \frac{1}{LR}\right) \\ \frac{YO}{2}\left(LF - \frac{1}{LR}\right) & \frac{1}{2}\left(LF + \frac{1}{LR}\right) \end{bmatrix}$$ $LF = 10^{LOSS(DB)/20} \quad LR = 10^{REVL(DB)/20}$

The GCP names for user-initiated sections are:
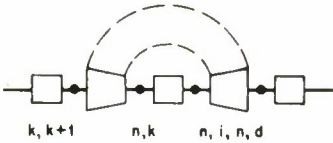
DUMA   DUMC

DUMB   DUMD

2. Operator Sections

SYMBOL	GCP NAME	GCP WILL REQUEST	TYPE OF SECTION
	SHUN JUNC	BR, SE	SHUNT connected JUNCTION to another branch
	SERI JUNC	BR, SE	SERIES connected JUNCTION to another branch
	PARB p		Junction nearer the load for initiating a PARALLEL BRANCH (numbered p) in PARALLEL connection with main branch (numbered n)
	COMB		JUNCTION nearer the generator for ReCOMBINING parallel branch (numbered p) in parallel connection with the main branch (numbered n). Note: $m - 1 > k$ Series connected junctions are handled as described in Sec. I-G-2.

13

SYMBOL	GCP NAME	GCP WILL REQUEST	TYPE OF SECTION
	IMAG n, d, n, i	START, END (BR, SE)	The section placed at n, k will be the mirror image of the cascade of sections between n, d and n, i. $k \geqslant i > d$

B. Measurement MENU

These measurements may be called only after giving the command MEAS (n and k are the two-digit branch and section numbers, respectively, of the terminal plane at which the measurement is to be initiated).

1. Driving Point Measurements — Absolute

These measurements may be made at any terminal plane.

n, k ZLOD	Impedance at terminal plane n, k looking toward decreasing section numbers.
n, k ZGEN	Impedance at terminal plane n, k looking toward increasing section numbers
n, k YLOD	Admittance at terminal plane n, k looking toward decreasing section numbers.
n, k YGEN	Admittance at terminal plane n, k looking toward increasing section numbers.

2. Driving Point Measurements — Ratios

These measurements must be made at a terminal plane between a pure resistance termination and the remainder of the circuit.

n, k REFL	Complex reflection coefficient (S11 or S22, if properly terminated)
n, k REFM	Magnitude of reflection coefficient
n, k RFPH	Phase of reflection coefficient in degrees
n, k VSWR	Voltage standing wave ratio
n, k RFLO	Reflection loss in decibels
n, k RTLO	Return loss in decibels

3. Transfer Measurements

These measurements require resistance terminations at both ends of the branch being analyzed. Measurements must be made at a terminal plane between a pure resistance termination and the remainder of the circuit. The terminal plane at which the measurement is called represents the generator end for that measurement.

n, k TRAN	Complex voltage transmission coefficient [S12 if measured at load end (n, 01), or S21 if measured at generator end (n, k); k > 01, if properly terminated at both ends].

14

n, k TRMG		Voltage transmission coefficient magnitude
n, k TRPH		Voltage transmission coefficient phase
n, k TRLO		Transducer loss in decibels
n, k SCAT		Scattering matrix elements (no other measurement may be made at the same time)

C. Measurement Definitions
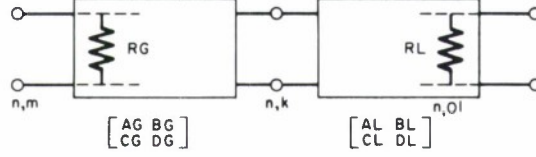
$$n, k \ \ ZLOD = \frac{AL}{CL} \qquad n, k \ \ YLOD = \frac{CL}{AL}$$

$$n, k \ \ ZGEN = \frac{DG}{CG} \qquad n, k \ \ YGEN = \frac{CG}{DG}$$

$$n, k \ \ REFL = \frac{\dfrac{AL}{CL} - \dfrac{DG}{CG}}{\dfrac{AL}{CL} + \dfrac{DG}{CG}}$$

$$n, 01 \ \ REFL = - \ \frac{\dfrac{AL}{CL} - \dfrac{DG}{CG}}{\dfrac{AL}{CL} + \dfrac{DG}{CG}}$$

$$n, k \ \ REFM = |REFL| \quad .$$

$$n, k \ \ RFPH = 2 \arctan \frac{Im \ REFL}{Re \ REFL + \sqrt{(Re \ REF)^2 + (Im \ REFL)^2}}$$

$$n, k \ \ VSWR = \frac{1 + REFM}{1 - REFM}$$

$$n, k \ \ RFLO = 10 \log_{10} \left| \frac{1}{1 - REFM^2} \right|$$

$$n, k \ \ RTLO = 20 \log_{10} \frac{1}{REFM}$$

$$n, k \ \ TRAN = \begin{cases} S12 & \text{if} \quad k = 01 \\ S21 & \text{if} \quad k > 01 \end{cases}$$

$$n, k \ \ TRAN = \frac{2 \left[\dfrac{DG}{CG} \cdot YLOD(n, 01) \right]^{1/2}}{AL + \dfrac{DG}{CG} CL} = \frac{2 \left[RG(n, k+1)/RL(n, 01) \right]^{1/2}}{A(n, k) + RG(n, k+1) \ C(n, k)}$$

$$n, 01 \ \ TRAN = \frac{2 \left[RL(n, 01)/RG(n, k+1) \right]^{1/2} \left[A(n, k) \ D(n, k) - B(n, k) \ C(n, k) \right]}{D(n, k) + RL(n, 01) \ C(n, k)}$$

$$n, k \ \ TRMG = n, k \ |TRAN|$$

$$n, k \ \ TRMG = n, 01 \ |TRAN|$$

15

$$n, k \ \text{TRPH} = 2 \arctan \frac{\text{Im TRAN}}{\text{Re TRAN} + \sqrt{(\text{Re TRAN})^2 + (\text{Im TRAN})^2}}$$

$$n, k \ \text{TRLO} = 20 \log_{10} \frac{1}{\text{TRMG}(n, k)}$$

$$n, 01 \ \text{TRLO} = 20 \log_{10} \frac{1}{\text{TRMG}(n, 01)}$$

$$n, k \ \text{SCAT}, \quad S11 = n, k \ \text{REFL} \qquad S12 = n, 01 \ \text{TRAN}$$
$$S21 = n, k \ \text{TRAN} \qquad S22 = n, 01 \ \text{REFL}$$

D. Commands

NORMAL COMMANDS

COMMAND	GCP WILL REPLY	FUNCTION – OPERATOR RESPONSE
GCP-MOD 2	Execution begins 01, 01	Initiates operation of GCP-MOD 2. Type in name of first section in first branch (e.g., SHUN RESI) or RESE (see below) if previous work is to be recalled.
SEMO	BR, SE, NEW SECT.	SECTION MODIFICATION. Type in branch and section numbers, and name of new section wanted.
ELMO	BR, SE, ELEM	ELEMENT MODIFICATION. Type in branch and section numbers, and name of parameter to be changed, e.g., 01, 05, ZO.
	NEW VALUE	Type in new value wanted.
BRAN n	n, k	Sends GCP to Branch indicated by n, the two-digit number of the branch. k is the two-digit number of the first section not previously used in that branch. Type in name of section wanted for n, k, or command for action on that branch.
MEAS	BR, SE, TYPE	Initiates MEASUREMENT. Type in branch and section numbers of terminal plane where measurement is to be made (terminal plane adjacent to assumed generator for transfer measurements) and names (up to three) of measurements to be made.
	FREQ(GHZ), MIN, MAX, INCR	Type in frequency range and increment.
TUNE	BR, SE, ELEM	TUNES (increments) value of one section parameter and makes single frequency performance measurements at each increment. Type in branch and section numbers and name of element to be tuned.

16

NORMAL COMMANDS (Continued)

COMMAND	GCP WILL REPLY	FUNCTION – OPERATOR RESPONSE
TUNE	BR, SE, MEAS	Type in branch and section numbers of terminal plane and names (up to three) of measurements to be made.
	ELEM(MIN, MAX, INCR.), FREQ(GHZ)	Type in range and increment for variation of element value, and frequency at which measurement is wanted.
OFFL	NEXT	Prints measured data in a file named GCP DATA. The CMS command offline print gcp data must be issued after terminating GCP-MOD 2.
		This command remains in effect until ONLI is issued.
ONLI	NEXT	Resume printing of measured data on the console.
RESE	n, k	<u>RESETS</u> GCP at next to last developed section. Usually used to reinstate data file after intended or unintended interruption in operation of GCP. Retype name of last section, or type command for other action wanted. Data file will be erased if RESE is not used in response to GCP printing 01, 01 at initiation of operation of GCP.
QUIT		Terminates operation of GCP, but does not erase data file.

DIAGNOSTIC COMMANDS*

COMMAND	GCP WILL REQUEST	FUNCTION
LIST LIST ALL	BR, or BR, SE	Prints out section name and element values for BR, SE specified, or if only BR is specified, prints out section names and element values for all sections in that branch, or if ALL is used in the initial command, prints out all section names and element values of all branches used in that session.
		Element values can be identified as to parameter involved by comparing order on print-out to order given below under LIST.
MATR MATR ALL	BR, or BR, SE	Prints out elements of transfer matrix, for the last frequency measured, for BR, SE specified; or if only BR is specified, prints out all elements of the section transfer matrices for that branch; or, if ALL is used in the initial command, prints out elements of all transfer matrices of all branches used in that session.

---

* These commands may be used at any point in a session with GCP to exhibit the data GCP has been working with.

SERI, RESI
  OHMS

SERI, CAPA
  FARADS

SERI, INDU
  HENRIES

SERI, TEMS
  OHMS
  LENG(IN.)
  $(DIEL)^{1/2}$

SERI, PRLC
  MHOS
  FARADS
  HENRIES

SERI, SRLC
  OHMS
  FARADS
  HENRIES

SERI, PRES
  $2\pi FO$
  B(MHOS)
  $1/Q$

SERI, SRES
  $2\pi FO$
  X(OHMS)
  $1/Q$

SERI, TEML
  OHMS
  LENG(IN.)
  $(DIEL)^{1/2}$

SERI, JUNC
  BR
  SE
  2

SERI, TEWS
  WIDE
  LENG(IN.)
  $(DIEL)^{1/2}$
  NARO

SHUN, RESI
  MHOS

SHUN, CAPA
  FARADS

SHUN, INDU
  HENRIES

SHUN, TEMS
  OHMS
  LENG(IN.)
  $(DIEL)^{1/2}$

SHUN, PRLC
  MHOS
  FARADS
  HENRIES

SHUN, SRLC
  OHMS
  FARAD
  HENRIES

SHUN, PRES
  $2\pi FO$
  B(MHOS)
  $1/Q$

SHUN, SRES
  $2\pi FO$
  X(OHMS)
  $1/Q$

SHUN, TEML
  OHMS
  LENG(IN.)
  $(DIEL)^{1/2}$

SHUN, JUNC
  BR
  SE
  1

SHUN TEWS
  WIDE
  LENG
  $(DIEL)^{1/2}$
  NARO

TEML
  OHMS
  LENG(IN.)
  $(DIEL)^{1/2}$

ISOL
  OHMS
  FORL = DB
  REVL = DB

ATTN
  OHMS
  DB

XFMR
  TURNS RATIO

INVE
  OHMS

RADL
  DGEN/2
  $(DIEL)^{1/2}$
  B = B
  DLOD/2

IMAG
  $\left.\begin{array}{l}BR\\SE\end{array}\right\}$ START
  $\left.\begin{array}{l}BR\\SE\end{array}\right\}$ END

PARB
  BR

WISO
  WIDE
  FORL = DB
  REVL = DB
  $(DIEL)^{1/2}$
  NARO

WLOD
  WIDE
  0
  $(DIEL)^{1/2}$
  NARO

III. PROGRAM DESCRIPTION

A. Introduction

GCP-MOD2 is a CP/CMS Fortran program designed to allow users without programming knowledge to perform circuit analysis on a wide range of circuits. The following description is somewhat of a verbal flow chart in which the flow of information supplied by the user is processed by GCP.

GCP has one general control program to which different circuit libraries can be attached. The circuit libraries are text libraries, containing subroutines written in a specified form, which convert information about the circuit supplied by the user into information processed by GCP.
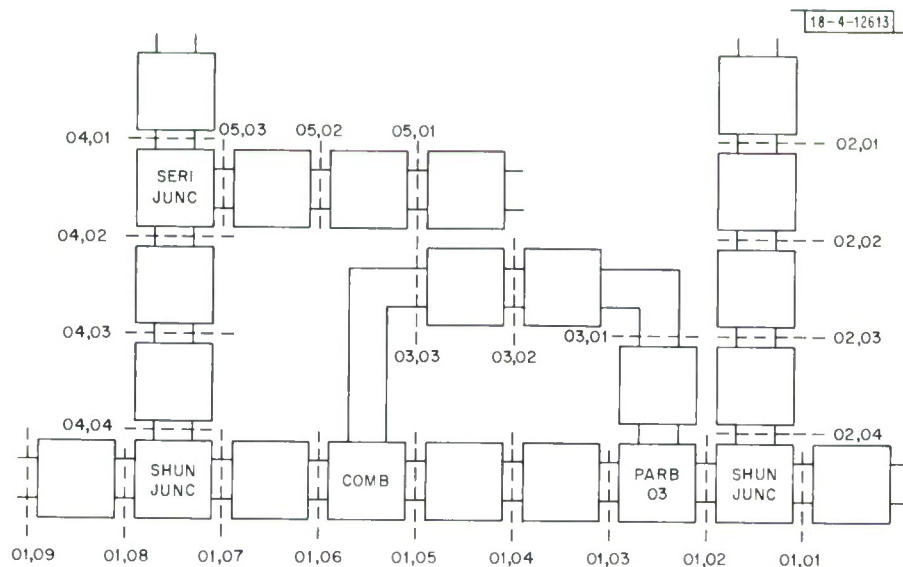
Fig. 2. Typical circuit layout.

Figure 2 shows schematically the way the circuits are laid out. The layout consists of up to ten branches with each branch capable of handling fifty sections. Each section contains a circuit element or elements, when more than one element can be conveniently represented in one section. Each section is converted into a transfer matrix. The sections are identified by branch and section number (for example, 03, 05 refers to the third branch, and the fifth section).

The control program, Fig. 3, consists of several executive modules which control the flow of information. These executive modules function automatically depending on the information given by the user. Information given by the user is stored in two arrays. One array, which contains the transfer matrices, is ABCD (10, 50, 4) and is dimensioned as shown, representing 10 branches, 50 sections, and 4 matrix elements. The array is complex. The second array, STOR (10, 50, 4, 2, 2), stores information pertaining to branch number, section number, element values, section type, and element type, respectively.

The STOR array is also written into the data file with data set reference number 4 for a permanent record of the information in case of system failure or fatal error on the user's part. The file name is GCP STORAGE.
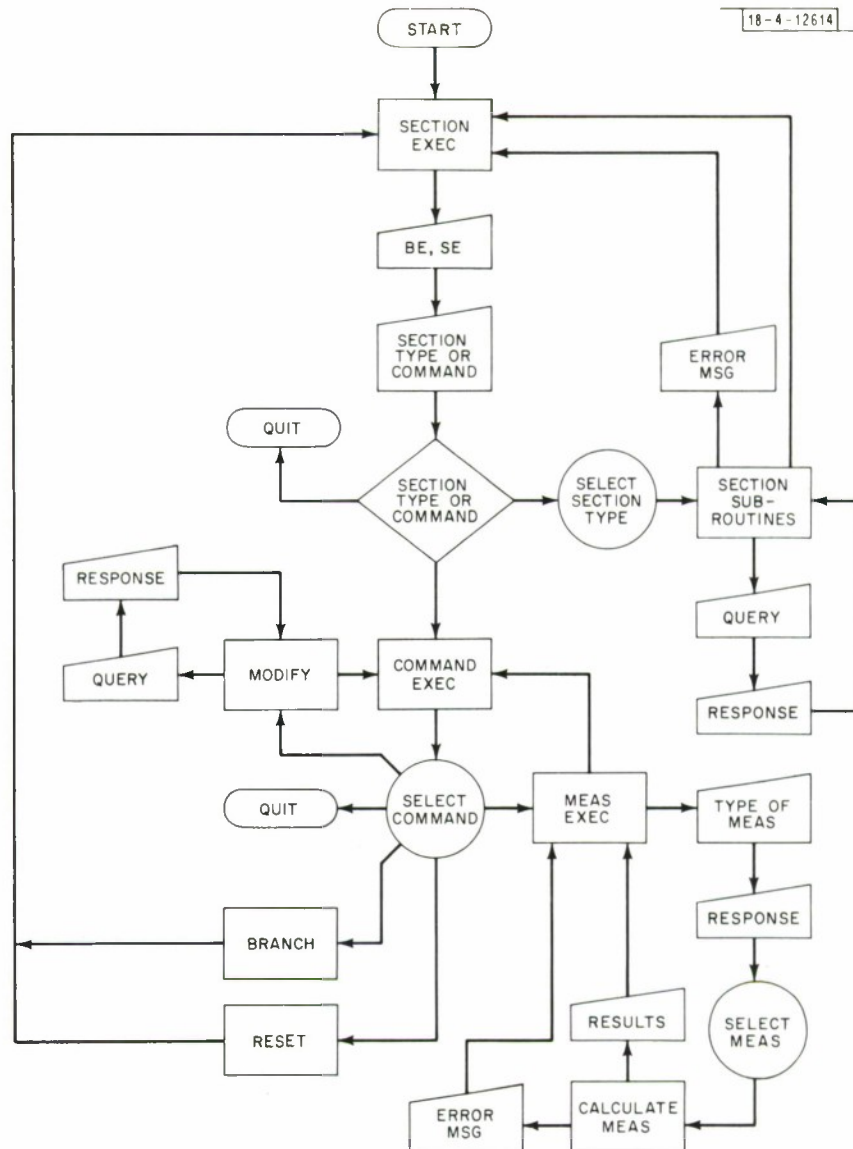
Fig. 3. Simplified GCP flow diagram.

These two arrays are in a labeled common named INDEX. There are several labeled commons which are needed to transfer information between subroutines.

An exec program is used to start GCP. It does a GLOBAL LOADER and a LOAD GCP (NOMAP XEQ), loading the appropriate text libraries and GCP. A LOAD MODULE is also available, and the program may be started by issuing GCP-MOD2. Once the program is started, all arrays are initialized to zero. This feature is used to test the arrays to find out how much of each is used. After initializing, the first executive module (section exec, Fig. 3) is entered. This module sets the branch and section number and checks to make sure the branch and section limits are not exceeded, then writes the data from the last section formed into GCP storage on the disk file, using a direct access write. The section exec then prints the next available branch and section number on the typewritter console. Present branch and section numbers are stored in the integer variables N and M, respectively.

The user types in a section type or a command and the information is read into two integer variables, TYPE and ELEM1 under an A4 format. The information supplied by the user is first compared to a list of commands using logical if statements. If a command was given, the appropriate logical if executes the command. If a section request is given, a subroutine SUB is called. This subroutine passes the arguments TYPE and ELEM1. The subroutine contains a list of section types, and the request is compared to the list of section types using logical ifs. If no match is found, an error is assumed and an error message is typed on the typewritten console. The branch and section numbers are reset and control is returned to the section exec where the branch and section number is typed out. The section exec is then ready for more input information. If a match is found, the appropriate section subroutine is called.

B. Section Subroutines

Section subroutines are generally set up in the following manner:

The subroutine name is the name of the section with a prefixed Q (i.e., TEML is QTEML). Generally no arguments are passed except one or more return statement numbers using * as defined in Fortran IV.

Complex statements for the various complex variables are required, along with the appropriate common statements. Labeled common INDEX is always required in all subroutines since it contains N, M and the STOR array. These variables and array are needed to identify the section. The labeled common ZMEAS is generally required because it contains section and branch information needed in the entry section.

Generally a NAMELIST is required since each section requires specific information to be supplied by the user. This is accomplished by a write statement which asks for the information and a NAMELIST READ which reads it.

Section information (length of a section of transmission line for example) is placed in the STOR array. Eight storage locations are available for this data. They are the third and fourth dimensions of the array.

Finally each section contains an identification code stored in STOR (N, M, 1, 2, 1). This code is used to recall the subroutine.

To this point, the subroutine has stored the information pertaining to a specific section, and control passes back to the section exec.

Each section subroutine contains an entry point. The entry name is the same as the subroutine name, but with a Z prefixed instead of Q. (i.e., TEML would be ZTEML). The entry is called for a measurement and generates the transfer matrix using information stored in the STOR array.

C. Commands

All of the remaining subroutines to be described pertain to commands. The commands are few in number and are: BRAN XX, RESE, SEMO, ELMO, MEAS, TUNE, LIST, MATR, and QUIT.

BRAN XX, where XX is a branch number, either starts a new branch if the branch is empty or enters the branch at the next available section. BRAN and the branch number (XX) are read into integer variables TYPE (or ACTION) and ELEM1 under an A4 format.

The list of commands is scanned using logical ifs and the appropriate if statement calls the subroutine QJUNC. QJUNC tests ELEM1 to be sure a branch number was given (i.e., tests to see if it is not blank). If it is blank an error message is typed out on the console and control is passed back to the user for appropriate action.

If no error is detected, ELEM1 must be changed from an A formatted variable to an I formatted variable. This is accomplished by writing it into a file with data reference set number 1 and reading it back as an I format.

Using a do loop, STOR (ELEM1, 1, 1, 2, 1) is first checked to see if it is zero (this location contains the program identification code). If it is zero, this indicates that it is an empty branch and therefore a new branch is being called. The section variable M is set to zero and control returned to the section exec and the user.

If STOR (ELEM1, 1, 1, 2, 1) is not zero, the do loop continues and cycles the second dimension until a zero is found indicating that this section is empty (the second dimension is the storage location containing section numbers). The section variable is set to this number and control is returned to the section exec, whereupon the section exec types the branch and section numbers at the console, and control passes to the user.

The command RESE allows the user to reset GCP. This command is usually given at the start of the program to recall all information into active storage. The command was implemented to allow the user to recall previously stored information in case of a system crash or fatal user error.

As previously stated, all information is written into file GCP STORAGE. The RESE command sets the file pointer to the top of the file and sequentially reads this direct access file into the STOR array. Since the information is read originally into GCP STORAGE, one section behind the present section, RESE resets to that section. This command must be given as the first command if a reset is desired. If it is not given, the GCP storage file is erased using CP/CMS CALL ERASE statement.

RESE may be given anytime during execution and the active storage will be reset to the previous section.

The SEMO command calls a subroutine which allows the user to modify or change a section. The subroutine asks for branch, section, and section type, then does a read similar to the section exec read for a section request.

STOR (N, M, 1, 2, 1) is checked for zero to see if a section exists. If STOR (N, M, 1, 2, 1) is zero, an error message is printed and control passed back to the command exec and the user. If STOR (N, M, 1, 2, 1) is not zero, a list of section types is scanned using logical ifs. If a match is found, the appropriate subroutine is called and the STOR array updated for that section.

In the event no match is found, an error message is printed and control is returned to the command exec and the user for corrective action.

If no errors are detected, file GCP STORAGE is updated using a direct access write to GCP STORAGE.

ELMO operates in a manner similar to SEMO execpt that only one element of the section is modified. It has the same safeguards as SEMO to insure that the proper branch, section, and element are to be modified.

ELMO contains a list of element types in logical if statement and these are scanned and the appropriate location of the STOR array is updated when a match is found. ELMO also updates GCP STORAGE as SEMO does.

D.   Diagnostic Commands

The LIST command calls subroutine QLIST which prints the circuit on the typewriter console. Do loops cycle through STOR (N, M, 1, 2, 1) (identification code) of each section and determine the section type by comparing this STOR array location to a list of subroutine identification codes using logical ifs. When a match is found the if action is a write statement which prints the section type.

At the start of each loop the do loop indices are printed as the branch and section numbers (N, M). The indices are integer variables and may be fixed by the user to display the contents of only one section. This is done in two ways.

If the user issues the commands LIST ALL, ALL is read into the integer variable ELEM1. This variable is used in QLIST using a logical if to set the do loop indices so that all sections and branches are cycled through.

If the user does not specify ALL, then a write statement asks for a branch or a branch and section. The branch and section is read into two integer variables and these are used as the do loop indices. If just a branch is given, the do loop indices are fixed so that the whole branch is cycled.

A nested do loop is used to display the values of the section elements (i.e., the third dimension of the STOR array). Here again, if zeroes are found, the do loop is terminated to prevent printing of zeroes for sections which have only one element.

The MATR command is similar to the LIST command except that the ABCD array is used instead of the STOR array, and the printing is different. It is handled in the same way as LIST as far as the do loops are concerned.

OFFL sets an integer variable NRITE to 8. All write statements which write measured data use NRITE as a data reference set number. When NRITE equals 8, all writes are direct access, and write on a file named GCP DATA. Thus after terminating the program an OFFLINE PRINT GCP DATA must be issued.

ONLI sets NRITE to 6, and subsequent writes are on the typewriter.

The final command to GCP is QUIT. This command terminates the execution of the program. This is accomplished by scanning the list of commands using logical ifs. When a match is found, the if action is STOP.

E.   Measurements

The MEAS command initiates the following complex series of actions.

The logical ifs are scanned for a match to MEAS. When it is found, control is passed to a section of GCP which first initializes the three measurement variables to blanks (up to 3 measurements can be called).

GCP reads the measurement into 3 integer variables, MEAS1, MEAS2, and MEAS3, and the branch and section into N1, M1 respectively. The variables are cleared to blanks to prevent any previously called measurement from being called again.

Section and branch checks are performed testing STOR (N, M, 1, 2, 1) for zero to insure a valid measuring point (i.e., no measurement can be made of a section which has not been formed). An error message is printed if an error is detected and control returned to the user for corrective action. If no error is detected, the subroutine MEASUR, which now acts as the measurement exec, is called, passing arguments MEAS1, MEAS2, and MEAS3.

Measurement exec first checks MEAS1, MEAS2, and MEAS3 to be sure they are not all blank using logical ifs. If they are all blank, an error message is printed and control is returned to command exec and then to the user for corrective action.

The frequency range is asked for with a write statement. A NAME LIST READ reads the frequency data and is checked to insure proper form (i.e., start frequency smaller than stop frequency and increment not equal to zero). If errors are detected, appropriate error messages are printed and control is passed back to the user for corrective action.

Integer variables NCHECK and NUMBER are set to zero. NCHECK is used to suppress titles (zero prints titles; 1 or 2 does not print titles). The NUMBER variable is a counter.

Logical ifs are used to determine how many measurements were called. This is accomplished by checking to see if MEAS2 or MEAS3 are blank. If more than two measurements were called (MEAS2 not blank) then the subroutine TITLE is called, passing the arguments MEAS1, MEAS2, and MEAS3.

This subroutine checks the list of measurements against the passed arguments using logical ifs, and writes the appropriate titles for the columns on the typewriter.

Control is passed back to measurement exec and the subroutine QCALCZ is called. This subroutine controls the generation of the transfer matrices. Using two nested do loops it cycles through the first and second dimension of the STOR array (these locations contain branch and section numbers). Each time it goes through the do loop it calls ZSUB (an entry of subroutine SUB), which checks the section routine identification codes using logical ifs. When a match is found, the "Z" entry of the section subroutine is called and this in turn generates the transfer matrix of that section and stores it in the ABCD array.

Control returns to ZSUB which returns to ZCALCZ, and the do loops cycle again for the next section. When all the sections to be measured have been cycled through and have transfer matrices, the do loop is exited. This is accomplished by checking STOR (N, M, 1, 2, 1) for zero as before.

All transfer matrices have now been generated, and next must be multiplied. The subroutine MATMUL is called to multiply each transfer matrix by the product of the previous matrices.

MATMUL is simply a small subroutine for multiplying complex matrices by the product of the previous matrices. This is accomplished with a do loop and temporary storage. The limits of the do loop are set by QCALCZ. The upper limit M1 being carried through from measurement exec is in labeled common FREQ. When the do loop is satisfied, control returns to QCALCZ and the final matrix elements are stored in the complex variables A, B, C, and D. The limits of the do loop (in QCALCZ) are reset to provide transfer matrices and calculations for the sections to the left of the measuring point back to the generator. This transfer matrix is required for some measurements. The elements of this matrix are stored in the complex variables AG, BG, CG, and DG.

The scattering matrix elements are also generated and stored in S11, S12, S21, and S22. Control then passes back to the measurement exec.

Depending on the number of measurements called, appropriate action is performed to correlate the titles and columns to the data to be printed. The subroutine SUBMEA is now called. This subroutine scans a list of measurements using logical if to find the measurement that has to be called. When a match is found, a call to the appropriate entry point in QMEAS is made.

QMEAS is a subroutine consisting mainly of entry points, each one of which performs calculations to obtain the measurement requested. The result of the calculation is stored in an array ARR, where even numbered subscripts contain the real part and odd numbered subscripts contain the imaginary part, or blank if the result is all real. This arrangement is necessary for printing purposes.

If only one measurement is requested, QMEAS will print the results before passing control to the measurement exec. If more than one measurement is called, control is passed back to MEASUR without printing. The variable NCHECK is used to test whether QMEAS does any writing.

If more than one measurement is requested, the measurement exec will print out the results of the calculation. The results of the calculations are in the labeled common MEA.

After printing the results the frequency is incremented and the whole process is repeated for the next frequency. When the stop frequency is reached, control is passed back to the command exec and the user.

I.   QSER

Program identification code — 2

Other subroutines called   — None

Type statements needed   — COMPLEX ABCD (50, 50, 4), RESULT (4)
                                  REAL LEN

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2),
                                 W, RESULT, ELEM1
                                 COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
            Statement No. 5 of QSEMO

ABCD Matrix

   SERI, RESI

        A = 1

        B = R   R is in ohms

        C = 0

        D = 1

   SERI, CAPA

        A = 1

        $B = \dfrac{1}{j\omega C}$ C is in pF

        C = 0

        D = 1

   SERI, INDU

        A = 1

        $B = j\omega L$ L is in nH

        C = 0

        D = 1

   SERI, TEMS

        A = 1

        $B = jZ_o \sin \theta / \cos \theta$

        C = 0

        D = 1

      where $\theta = \omega l \sqrt{\epsilon\mu}/11.80271 \times 10^{9}$

             l = length in inches

             $\epsilon\mu$ = dielectric constant

   SERI, TEML

        A = 1

        $B = -jZ_o \, \text{eotan} \, \theta$

        C = 0

        D = 1

      where θ is as above

SERI, PRLC

    $A = 1$

    $B = 1/R + j(\omega C - 1/\omega L)$ R, L, C as above

    $C = 0$

    $D = 1$

SERI, SRLC

    $A = 1$

    $B = R + j(\omega L + 1/\omega C)$ R, L, C as above

    $C = 0$

    $D = 1$

SERI, PRES

    $A = 1$

    $B = BP \dfrac{1}{[1/Q + j(F/F_0 - F_0/F)]}$ F = frequency
    F_0 = resonant frequency
    BP = susceptance

    $C = 0$

    $D = 1$

SERI, SRES

    $A = 1$

    $B = X [1/Q + j(F/F_0 - F_0/F)]$ F, F_0, BP as above

    $C = 0$

    $D = 1$

QSER is a subroutine which handles all series elements. The element type is passed to QSER stored in the integer variable ELEM1 through the labeled common INDEX. Using a list of logical ifs an appropriate branch is made to a part of the subroutine which asks for the information pertaining to the element type and stores it in the STOR array.

The storage location STOR (N, M, 1, 1, 2) is coded for use when generating the transfer matrix.

II.   QSHU

   Program identification code — 3

   Other subroutines called   — None

   Type statements needed   — COMPLEX ABCD (10, 50, 4), RESULT (4).
                                      REAL LEN

   Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2),
                                     W, RESULT, ELEM1
                                     COMMON/ZMEAS/L, L1

   Returns — Statement No. 1 of SUB
               Statement No. 5 of QSEMO

   ABCD Matrix

      SHUN, RESI

          $A = 1$

          $B = 0$

          $C = R$ R is in ohms

          $D = 1$

SHUN, CAPA

    $A = 1$

    $B = 0$

    $C = j\omega C$ C is in pF

    $D = 1$

SHUN, INDU

    $A = 1$

    $B = 0$

    $C = \dfrac{1}{j\omega L}$ L is in nH

    $D = 1$

SHUN, TEMS

    $A = 1$

    $B = 0$

    $C = \dfrac{\cos\theta}{jZ_o \sin\theta}$

    $D = 1$

    $\theta = \dfrac{\omega l \sqrt{\epsilon\mu}}{11.80271 \times 10^9}$

  where l = length in inches

    $\epsilon\mu$ = dielectric constant

SHUN, TEML

    $A = 1$

    $B = 0$

    $C = \dfrac{\tan\theta}{jZ_o}$

    $D = 1$

  where θ and $\epsilon\mu$ are as above

SHUN, PRLC

    $A = 1$

    $B = 0$

    $C = R + j(\omega C - \dfrac{1}{\omega L})$ R, C, L as above

    $D = 1$

SHUN, SRLC

    $A = 1$

    $B = 0$

    $C = \dfrac{1}{R} + j(\omega L + \dfrac{1}{\omega C})$ R, C, L as above

    $D = 1$

SHUN, PRES

    $A = 1$

    $B = 0$

    $C = BP \left[\dfrac{1}{Q} + j\left(\dfrac{F}{F_o} - \dfrac{F_o}{F}\right) \right]$

    $D = 1$

SHUN, SRES

$$A = 1$$
$$B = 0$$
$$C = \frac{1}{X \, [1/Q + j(F/F_o - F_o/F)]}$$
$$D = 1$$

QSHU is a subroutine which handles all shunt elements, and is identical to QSER except for the generation of the transfer matrices.

III. QTEML

Program identification code — 4

Other subroutines called — None

Type statements needed — COMPLEX ABCD (10, 50, 4)
REAL LEN

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2), W
COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

$$A = \cos \theta$$
$$B = jZ_o \sin \theta$$
$$C = \frac{1}{jZ_o \sin \theta}$$
$$D = \cos \theta$$

where $\theta = \omega l \sqrt{\epsilon\mu}/11.80271 \times 10^9$

l = length in inches

$\epsilon\mu$ = dielectric constant

This subroutine simulates a length of transmission line. Z_o, length and dielectric constant are read into variables Z0, LEN, and EPMU, then stored in the STOR array.

IV. QSHORT

Program identification code — 5

Other subroutines called — None

Type statements needed — COMPLEX ABCD (10, 50, 4)

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2), W
COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

$$A = 0$$
$$B = 1$$
$$C = 1$$
$$D = 0$$

This subroutine simulates a short circuit at a branch end. The user does not supply any information to this subroutine.

30

V. QJUNC

Program identification code — 6

Other subroutines called — ZZSUB, Z1SUB, MATMUL

Type statements needed — COMPLEX ABCD (10, 50, 4), RESULT (4),
          ZIN, YIN
          INTEGER SER, SHU, CONN, QUIT,
          BLK, ELEM1, TYPE

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2),
          W, RESULT, ELEM1
          COMMON/ZMEAS/L, L1
          COMMON/FREQ/STAF, STOF, DELF, M1, N1, N2,
          N3, M2, M3

Returns — Statement No. 1 of SUB
    Statement No. 5 of QSEMO

ABCD Matrix

  SHUNT JUNCTION

    $A = 1$

    $B = 0$

    $C = Y_{in}$ Y_{in} is C/A of the product of transfer matrices of the specified branch.

    $D = 1$

  SERIES JUNCTION

    $A = 1$

    $B = Z_{in}$ Z_{in} is A/B of the product of transfer matrices of the speeified braneh.

    $C = 0$

    $D = 1$

This subroutine is used to attach SHUNT or SERIES branches to the main branch, or to another branch.

The subroutine has three sections. The first seetion deals with the mechanics of requesting new branehes and entering a branch at the next available section.

The seeond and third sections deal with the attaehing of branches to other branehes. The third seetion is actually a separate subroutine whieh is required because multiple calls to the same subroutine are not possible.

Branch and section numbers are stored in STOR (N, M, 1, 1, 1) and STOR (N, M, 2, 1, 1) respectively. Logical ifs test the variable type (containing the alphanumeric SHUN or SERI) to determine the connection of the branch (SHUNT or SERIES). The constant 1 is stored in STOR (N, M, 3, 1, 1) if a SHUNT connection is desired, and 2 is stored if a SERIES connection is desired.

To generate the transfer matrix for this section, it is necessary to call the subroutine ZZSUB which performs the same function as ZSUB deseribed in Sec. IV-E. This subroutine generates the transfer matrices for the seetion of the braneh to be attaehed. When all the transfer matriees have been formed, MATMUL is ealled to obtain the product of the matrices. Then the appropriate transfer matrix is generated for the junetion.

To attach a branch to a branch, subroutines ZZJUNC and Z1SUB are used as above for a single branch connection.

VI. QEMPTY

Program identification code — 7

Other subroutines called — None

Type statements needed — COMPLEX ABCD (10, 50, 4)

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2)
COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

$$A = 1$$
$$B = 0$$
$$C = 0$$
$$D = 1$$

This subroutine consists of just an identity matrix which simulates an empty section.

VII. QISOL

Program identification code — 8

Other subroutines called — None

Type statements needed — REAL LF, LR
COMPLEX ABCD (10, 50, 4)

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2)
COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

$$A = (x + y)$$
$$B = Z_O (x - y)$$
$$C = \frac{1}{Z_O (x - y)}$$
$$D = (x + y)$$

where $x = \frac{10^{LF/20}}{2}$ LF is forward loss (dB)

$y = \frac{1}{2 (10^{LR/20})}$ LR is reverse loss (dB)

This subroutine simulates an isolator. The forward and reverse loss in dB is read into LF and LR, then stored into STOR (N, M, 2, 1, 1) and STOR (N, M, 3, 1, 1) as 10.** (LF/20) and 10.** (LR/20), respectively. ZO is stored in STOR (N, M, 1, 1, 1).

VIII. QATTN

Program identification code — 9

Other subroutines called — None

Type statement needed — REAL L
COMPLEX ABCD (10, 50, 4)

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2)
COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

$$A = (x + y)$$

$$B = Z_O (x - y)$$

$$C = \frac{1}{Z_O (x - y)}$$

$$D = (x + y)$$

$$\text{where } x = \frac{1}{2} [10^{(L/20)}] \qquad L = \text{Attenuation (dB)}$$

$$y = \frac{1}{2 [10^{(L/20)}]}$$

This subroutine simulates a matched, fixed, attenuator. The attenuation is read into L then stored into STOR $(N, M, 2, 1, 1)$ as $10.^{**} (L/20)$. Z_O is stored in STOR $(N, M, 1, 1, 1)$.

IX. QXFMR

Program identification code — 10

Other subroutines called — None

Type statements needed — COMPLEX ABCD (10, 50, 4)

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2)
COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

$$A = N$$

$$B = 0$$

$$C = 0$$

$$D = \frac{1}{N}$$

This subroutine simulates an ideal transformer. The turns ratio is read into TURNS then stored in the STOR array.

X. QINVE

Program identification code — 11

Other subroutines called — None

Type statements needed — COMPLEX ABCD (10, 50, 4)

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2)
COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

$$A = 0$$

$$B = Z_O$$

$$C = \frac{1}{jZ_O}$$

$$D = 0$$

This subroutine simulates an impedance inverter. Z_O is read into ZO then stored in the STOR array.

XI. QCROS

Program identification code — 12

Other subroutines called — None

Type statements needed — COMPLEX ABCD (10, 50, 4)

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2)
COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

$$A = -1$$
$$B = 0$$
$$C = 0$$
$$D = -1$$

This subroutine simulates a change in polarity. No information is required from the user.

XII. QRADL

Program identification code — 13

Other subroutines called — BESJ, BESY

Type statements needed — REAL J1X, J0X, J1Y, J0Y
COMPLEX ABCD (10, 50, 4)

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2), W
COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

$$A = \frac{\pi Y}{2} [J1(Y) \ Y0(X) - Y1(X) \ J0(X)]$$

$$B = \frac{j\pi Y}{2} [J0(Y) \ Y0(X) - Y0(Y) \ J0(X)]$$

$$C = \frac{-j\pi Y \epsilon \mu R}{60B} [J1(Y) \ Y1(X) - Y1(Y) \ J1(X)]$$

$$D = \frac{\pi Y R}{2R_O} [Y0(Y) \ J1(X) - J0(Y) \ Y1(X)]$$

where $X = \dfrac{\sqrt{\epsilon\mu} \ \omega R}{11.8 \times 10^9}$; $Y = \dfrac{X R_O}{R}$

$$Z_O = \frac{60B}{R_O \sqrt{\epsilon\mu}}$$

B is the spacing

R is radius at the generator end

R_O is radius at the load end

$\epsilon\mu$ is the dielectric constant

J1, J0, Y1, Y0 are BESSEL functions

This subroutine simulates a radial line. The diameter of the load and generator ends are read into R_O and R, respectively. The spacing and dielectric constant are read into B and EPMU, respectively. STOR (N, M, 1, 1, 1) contains R/2., STOR (N, M, 2, 1, 1) contains EPMU**.5, STOR (N, M, 3, 1, 1) contains B, and STOR (N, M, 4, 1, 1) contains R_O/2.

The entry ZRADL calculates X and Y as above and calls the various BESSEL subroutines.

The BESSEL subroutines return an error flag and this is printed if an error occurs. The following table indicates the errors:

Error Code	Error
2	X is negative
3	Accuracy not obtained (set at one percent); or returned argument $>10^{70}$.

XIII. QDUMM

Program identification code — QDUMMA — 14
  QDUMB — 15
  QBUMC — 16
  QDUMD — 17

Other subroutines called   — None

Type statements needed   — COMPLEX (10, 50, 4)

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2), W
  COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
  Statement No. 5 of QSEMO

ABCD Matrix

  As required by user.

This subroutine consists of four entries which the user may use to write his own programs.

XIV. QIMAG

Program identification code — 18

Other subroutines called   — None

Type statements needed   — COMPLEX ABCD (10, 50, 4), TEMP (4), A, B, C, D

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2)
  COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
  Statement No. 5 of QSEMO

ABCD Matrix

  Product of several matrices.

This subroutine simulates a mirror image of specified sections. Start and end branch and section numbers are read into NS, MS, NE, and ME, then stored in the STOR array.

The transfer matrix is generated as in MATMUL, with the product stored in the ABCD array.

XV. QTEWG

QTEWG is an entry point in QTEML.

Program identification code — 4

Other subroutines called   — None

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

$$A = \cos \Theta$$

$$B = jZ_O \sin \Theta$$

$$C = \frac{1}{jZ_O \sin \Theta}$$

$$D = \cos \Theta$$

where

$$\Theta = \frac{2\pi \sqrt{(FREQ1^2 - FC^2)} \, (LEN) \sqrt{\epsilon\mu}}{11.80271 \times 10^9}$$

$$FREQ1 = \frac{\omega}{2\pi}$$

$$FC = 11.80271 \times 10^9 / [2(WIDE) \sqrt{\epsilon\mu}]$$

WIDE = wide dimension of waveguide

$$Z_O = \frac{120\omega \, (NARO)}{WIDE \sqrt{\epsilon\mu} \sqrt{(FREQ1^2 - FC^2)}}$$

NARO = Narrow dimension of waveguide

XVI. QWLOD

QWLOD is an entry point in QTEML.

Program identification code — 20

Other subroutines called — None

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

$$A = 1$$

$$B = 0$$

$$C = Y_O$$

$$D = 1$$

where:

$$Y_O = \frac{(WIDE) \sqrt{\epsilon\mu} \sqrt{(\frac{\omega}{2\pi})^2 - FC^2}}{120\omega \, (NARO)}$$

NARO = narrow dimension of waveguide

WIDE = wide dimension of waveguide

XVII. QWISO

QWISO is an entry point in QISOL.

Program identification code — 21

Other subroutines called — None

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

36

ABCD Matrix

$$A = 1/2(FL + \frac{1}{RL})$$

$$B = \frac{Z_o}{2(FL - \frac{1}{RL})}$$

$$C = \frac{Y_o}{2(FL - \frac{1}{RL})}$$

$$D = 1/2(FL + \frac{1}{RL})$$

where

$$FL = 10^{(\frac{\text{forward loss}}{20})}$$

$$RL = 10^{(\frac{\text{reverse loss}}{20})}$$

$$Z_o = \frac{120\omega(NARO)}{(\text{wide})\sqrt{\epsilon\mu}\sqrt{(\frac{\omega}{2\pi})^2 - FC^2}}$$

$$Y_o = \frac{1}{Z_o}$$

$$FC = \frac{11.80271 \times 10^9}{2(\text{WIDE})\sqrt{\epsilon\mu}}$$

WIDE = wide dimension of waveguide

NARO = narrow dimension of waveguide

XVIII. QPARB

Program identification code — 22

Other subroutines called — None

Type statements needed — COMPLEX ABCD (10, 50, 4)
INTEGER ELEM1

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2),
W, RESULT, ELEM1
COMMON/ZMEAS/L, L1

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix

A = 1
B = 0
C = 0
D = 1

This subroutine signals the start of a parallel branch.

The command is PARB p where p is a branch number. The branch number is read into the integer variable ELEM1 in an A4 format. It is then written on data reference set number 1 in an A4 format and read back as an I2 format. This is required to convert it from an A format to an I format.

The branch number is stored in STOR (N, M, 1, 1, 1) to be used later in the QCOMB subroutine.

With logical ifs the branch number is checked to make sure that a branch number was given, and that it is between 1 and 10.

The ABCD matrix is set to the unit matrix because the program computes the products of the ABCD matrix of the section when a measurement is called for.

XIX. QCOMB

Program identification code — 23

Other subroutines called — MATMUL
Z1SUB

Type statements needed — COMPLEX ABCD (10, 50, 4), RESULT (4),
TEMP (4), DEL1, DEL, ZDEL, Z11, ZZ11,
Z12, ZZ12, Z21, ZZ21, Z22, ZZ22

Common statements needed — COMMON/INDEX/M, N, ABCD, STOR (10, 50, 4, 2, 2)
W, RESULT
COMMON/ZMEAS/L, L1
COMMON/FREQ/STAF, STOF, DELF, M1, N1, N2,
N3, M2, M3

Returns — Statement No. 1 of SUB
Statement No. 5 of QSEMO

ABCD Matrix (both paths not pure shunt, see below)

$$A = \frac{B'A'' + B''A'}{B' + B''}$$

$$B = \frac{B'B''}{B' + B''}$$

$$C = C' + C'' + A'\left(\frac{D'' - D'}{B' + B''}\right) + A''\left(\frac{D' - D''}{B' + B''}\right)$$

$$D = 1 - \left[\frac{B'(1 - D'') + B''(1 - D')}{B' + B''}\right]$$

where:

A', B', C', D' are the elements of the matrix formed by multiplying the transfer matrices of one parallel branch.

A", B", C", D" are the elements of the matrix formed by multiplying the transfer matrices of the other parallel branch.

ABCD Matrix (both branches are pure shunt)

A = 1

B = 0

C = C' + C"

D = 1

where C', C" are as above.

QCOMB performs the function of creating transfer matrices for both branches, finding their products, and combining them into a single transfer matrix.

The subroutine first saves the integer variables M1, M2, M3, N1, N2, and N3, because the original values are destroyed.

Using a do loop the main branch is searched to determine the number of parallel sections it contains. This is done by checking STOR (L, LI, 1, 2, 1) for 22, which is the program identification code for PARB indicating the start of the parallel branch.

After this is determined the subroutine MATMUL is called to multiply the transfer matrices of these sections, and get product matrix. The integer variables M2, M3 are set to the proper values so that MATMUL multiples just the sections that are in parallel with the other branch.

38

The other parallel branch is then searched to determine the number of sections by checking STOR (10, 50, 4, 2, 2) for zeroes in a do loop.

The subroutine Z1SUB is called to generate the transfer matrices for these sections, and MATMUL is called to find their product.

At this point the product matrix of both branches is checked to determine if both branches are pure shunt. If both branches are pure shunt the B element is zero; checking the ABCD matrix above, it can be seen that a zero divide exception will occur if this matrix is generated. Using a logical if, the B element of each branch is checked for zero, and if they are not both zero, the variables M1, N1, M2, M3, N2, N3 are returned to their original values and the matrices are combined as in the first ABCD matrix.

If the B elements are both zero the matrices are combined as in the second ABCD matrix above.

Since the matrices of the main branch are multiplied to form a product matrix from which the measurements are made, it is necessary to change the sections of the main branch which are in parallel with the second branch to unit matrices. This is done using a do loop.

DOCUMENT CONTROL DATA – R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*	2a. REPORT SECURITY CLASSIFICATION
Lincoln Laboratory, M.I.T.	Unclassified
	2b. GROUP
	None

3. REPORT TITLE

Automatic Microwave Circuit Analysis With GCP-MOD 2

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Technical Note

5. AUTHOR(S) *(Last name, first name, initial)*

Getsinger, William J.
Maggiacomo, Thomas E.

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
10 July 1969	44	4

8a. CONTRACT OR GRANT NO. AF 19 (628)-5167	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO. 649L	Technical Note 1969-34
c.	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*
d.	ESD-TR-69-204

10. AVAILABILITY/LIMITATION NOTICES

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
None	Air Force Systems Command, USAF

13. ABSTRACT

GCP-MOD 2 is an extremely rapid means for performing frequency analyses of equivalent circuits of cascaded two-ports. An operator describes his circuit to GCP-MOD 2, using an abbreviated plain English vocabulary, and specifies measurements to be made at an appropriate terminal plane for frequencies of interest. GCP communicates laconically with the operator as it accepts the circuit description and any other instructions. It then tabulates the results of the measurements, and awaits further instructions, such as circuit modifications or different measurements.

As a typical example, a microwave filter circuit, made up of lengths of transmission line and five resonators, of both lumped and distributed elements, can be described to GCP and evaluated at a dozen frequencies for such measurements as insertion loss, transmission phase-shift, and input impedance, all in about 10 minutes.

14. KEY WORDS

frequency analyzer
microwave circuit analysis

GCP-MOD 2
computer